НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №4

з дисципліни **«**Комп’ютерне моделювання**»**

Виконав:

студент 3 курсу

ФІОТ гр. ІО-31

Долинний Олександр

Перевірив:

Марковський О.П.

Київ – 2015 р.

**Завдання:**

Спеціалізована обчислювальна система містить **6** вузлів, інтенсивність відмов яких дорівнює, відповідно **1\*10-5, 1\*10-4, 5\*10-4, 1\*10-3, 2\*10-3, 5\*10-2** год-1, вартість вузлів становить, відповідно **2, 3, 4, 4, 5, 5** тисяч гривень. Є ресурс у **12** тисяч гривень, які можна витратити на дублювання вузлів (в випадку дублювання відмовою вважається поломка всіх вузлів даного типу). Віднайти оптимальний варіант дублювання.

**Лістинг програми**

**Model.java**

**package km.lab4;**

**public class Model {**

**double[] intensArray;**

**double time;**

**int[] cost;**

**int totalCost;**

**int[] optX;**

**double optimumReliability;**

**double[][] functionArray;**

**int[][] xArray;**

**public Model(double[] intensArray, double time, int[] cost, int totalCost) {**

**this.intensArray = intensArray;**

**this.time = time;**

**this.cost = cost;**

**this.totalCost = totalCost;**

**}**

**void solve() {**

**functionArray = new double[totalCost + 1][cost.length];**

**xArray = new int[totalCost + 1][cost.length];**

**for (int i = 0; i <= totalCost; i++) {**

**xArray[i][0] = i / cost[0];**

**functionArray[i][0] = probWorking(intensArray[0], time, xArray[i][0] + 1);**

**}**

**for (int i = 1; i < cost.length; i++) {**

**for (int j = 0; j <= totalCost; j++) {**

**double optimumFunc = 0;**

**int optimumX = 0;**

**for (int k = 0; k <= j; k++) {**

**int tempX = k / cost[i];**

**double tempFunc = functionArray[j - tempX \* cost[i]][i - 1] \* probWorking(intensArray[i], time, tempX + 1);**

**if (tempFunc > optimumFunc) {**

**optimumFunc = tempFunc;**

**optimumX = tempX;**

**}**

**}**

**xArray[j][i] = optimumX;**

**functionArray[j][i] = optimumFunc;**

**}**

**}**

**optX = new int[cost.length];**

**optimumReliability = functionArray[totalCost][cost.length - 1];**

**optX[optX.length - 1] = xArray[totalCost][cost.length - 1];**

**int tempCost = totalCost - optX[optX.length - 1] \* cost[optX.length - 1];**

**for (int i = optX.length - 2; i >= 0; i--) {**

**optX[i] = xArray[tempCost][i];**

**tempCost -= optX[i] \* cost[i];**

**}**

**}**

**String getResult() {**

**String result = "Optimum reliability = " + optimumReliability + ";";**

**result += "\n";**

**for (int i = 0; i < cost.length; i++) {**

**result = result + " x" + (i + 1) + " = " + optX[i] + ";";**

**}**

**return result;**

**}**

**String getMatrix() {**

**String result = "";**

**for (int i = 0; i <= totalCost; i++) {**

**result = result + i + ":";**

**for (int j = 0; j < cost.length; j++) {**

**result = result**

**+ " F" + (j + 1) + " = " + String.format(" %.6f", functionArray[i][j])**

**+ "; x" + (j + 1) + " = " + xArray[i][j] + " |";**

**}**

**result += "\n";**

**}**

**return result;**

**}**

**double probWorking(double lambda, double time, int count) {**

**double result = 1 - Math.pow((1 - Math.exp(-lambda \* time)), count);**

**return result;**

**}**

**}**

**Main.java**

**package km.lab4;**

**public class Main {**

**public static void main(String[] args) {**

**//int[] cost = {3,4,3,6,5,4};**

**//double[] intens = {0.001,0.001,0.001,0.01,0.001,0.005};**

**int[] cost = {2,3,4,4,5,5};**

**double[] intens = {0.00001,0.0001,0.0005,0.001,0.002,0.05};**

**double time = 10;**

**int totalCost = 12;**

**Model model = new Model(intens,time,cost,totalCost);**

**model.solve();**

**System.out.println(model.getMatrix());**

**System.out.println(model.getResult());**

**}**

**}**